Mimic The Raw Domain: Accelerating Action Recognition in the Compressed Domain

Barak Battash
Intel
Haifa, Israel
barak.battach@intel.com

Haim Barad
Intel
Haifa, Israel
haim.barad@intel.com

Hanlin Tang
Intel Labs
San Francisco, CA
hanlin.tang@intel.com

Amit Bleiweiss
Intel
Haifa, Israel
amit.bleiweiss@intel.com

Abstract

Video understanding usually requires expensive computation that prohibits its deployment, yet videos contain significant spatiotemporal redundancy that can be exploited. In particular, operating directly on the motion vectors and residuals in the compressed video domain can significantly accelerate the compute, by not using the raw videos which demand colossal storage capacity. Existing methods approach this task as a multiple modalities problem. In this paper we are approaching the task in a completely different way; we are looking at the data from the compressed stream as a one unit clip and propose that the residual frames can replace the original RGB frames from the raw domain. Furthermore, we are using teacher-student method to aid the network in the compressed domain to mimic the teacher network in the raw domain. We show experiments on three leading datasets (HMDB51, UCF1, and Kinetics) that approach state-of-the-art accuracy on raw video data by using compressed data. Our model MFCD-Net outperforms prior methods in the compressed domain and more importantly, our model has 11X fewer parameters and 3X fewer Flops, dramatically improving the efficiency of video recognition inference. This approach enables applying neural networks exclusively in the compressed domain without compromising accuracy while accelerating performance.

1. Introduction

Video traffic is projected to capture 82% of all Internet traffic by 2022 [1]. Video understanding with deep neural networks portends to be an important future workload. However, the spatiotemporal redundancies intrinsic in videos complicate the deployment of such models. State-of-the-art approaches are typically two-stream convolutional neural networks [2] that require expensive computations of optical flow. Some works [3] tried to save the optic flow computation by using the motion vectors from the compressed stream as a coarse approximation of optic flow.

Video signals have significant redundancy in the spatial and temporal axis providing significant compression. This redundancy makes video inference and computation a very expensive task in deep learning. The goal of this paper is to find a more efficient way infer actions from videos with a minimal decrease in accuracy.

Performing deep learning on video tasks in the compressed domain is a new task presented by Wu *et al.*[4]. Inference in the compressed domain can save enormous amounts of storage, needed to store full RGB video clip and still infer with high accuracy. Furthermore, it would much more efficient to do the computation directly on the compressed **bit** stream; this would save capacity, computation and more importantly, the video decoding time and hardware. This is a very difficult task. Our domain as in previous works [5, 4, 6] is parallel to working in the partial compressed domain which will be named simply the compressed domain. Although our work is not on the compressed bit stream, this does take a step forward to the direction of working on a more compact representation.

In the compressed domain, there are three different components *I-frames*, residual frames and motion frames (motion vectors and residuals extracted from frames named *P-frames*). Recent attempts to accelerate such models in the compressed domain [5, 4, 6] used the same idea of modeling three different networks, one for each modality mentioned above. Their work still has a significant gap from the per-

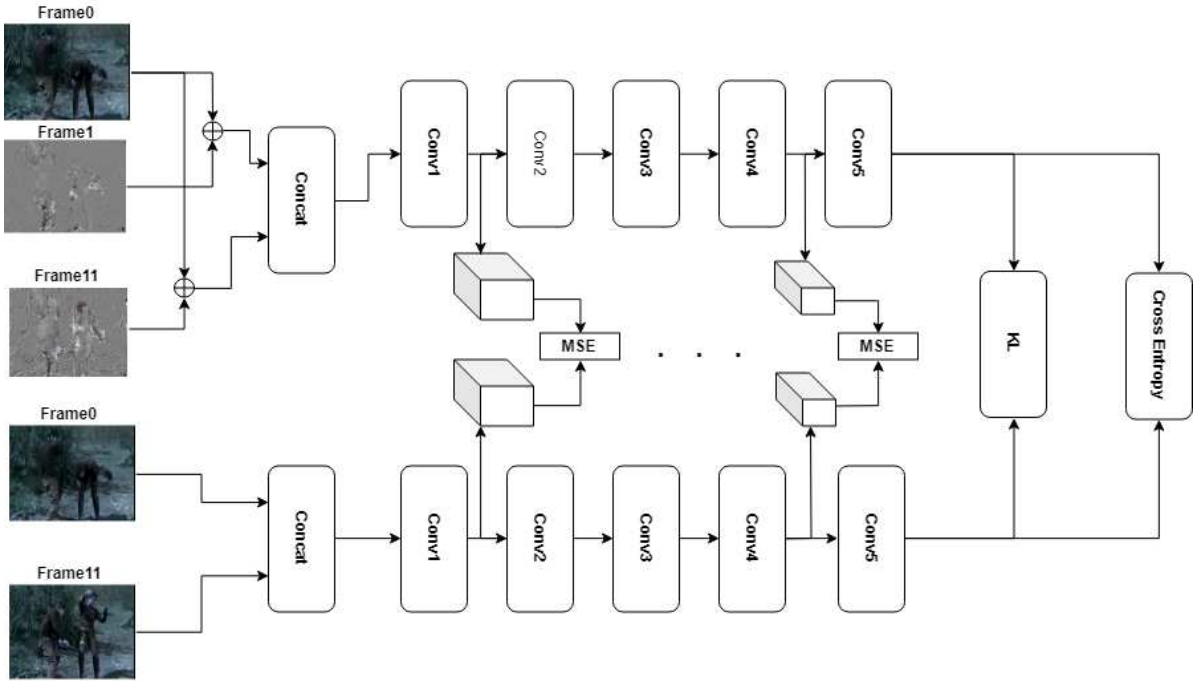

Figure 1. Overview of the proposed method showing the student-teacher framework where the student network is the network that is fed with a compressed clip which means 1*I-frame* and N augmented residual frames (N=11 in the figure) which can be seen in the upper part of the figure, below is the teacher network which is a pretrained network in the raw domain, we enforcing the compressed domain network $\Phi_{student}$ to mimic $\Phi_{teacher}$ by imposing $\Phi_{student}$'s intermediate feature maps to be similar to those of $\Phi_{teacher}$, by using MSE loss and KL loss on the logits

formance achieved by operating on the uncompressed RGB frames. In addition, these approaches require employing multiple 2D CNNs for various information components.

We approach this task differently than previous research. Instead of thinking of the residual and the motion vectors as different modalities, we claim that the residual frames are able to replace the missing RGB frames using a few adjustments in the video clip. We leverage the Multi-Fiber Net [7] as a 3D convolutional backbone and treat the video clip the same as a video in the raw domain, which holds N RGB frames. In the compressed domain we have one *I-frame* and N-1 *P-frames* and we will use those N-1 *P-frames* as a replacement for the missing RGB frames, this clip will be processed together using a 3D CNN. Our method brings dramatic reductions in memory and computational complexity while increasing accuracy. Compared to previous works in the compressed domain [4, 5], our method uses one 3D CNN and teacher-student framework in order to achieve the state of the art performance. This way of interfacing with the task help our network in the compressed domain, MFCD-Net to be more similar to the network in the raw domain MFNet[7].

We summarize our contributions as follows:

- We introduce a novel approach to accelerate the inference of action recognition in the compressed domain. Our method looks at the compressed components as a single unit, hence we are able to use a single 3D CNN.

- We approach the compressed stream as a pseudo video stream helping us to train the network using a student-teacher framework, which for the first time being done between the compressed domain and the uncompressed domain.

- Unlike previous work, our approach uses only the *I-frame* and the residuals but not the motion vectors. This leads to memory capacity and bandwidth decrease and computation savings.

- Compared to previous approaches in the compressed domain [5, 4, 6], our MFCD-Net beats the state of the art by 4.1 points on HMDB51[8] top1 and 2.3 point on UCF101 [9] top1, while requiring $11\times$ fewer parameters and $3\times$ fewer FLOPs.

- Our approach leads to a minimal decrease in accuracy across two leading datasets: UFC101 [9] and HMDB51 [8]. For example, our decrease in accuracy is only 0.9 on the UCF101 dataset.

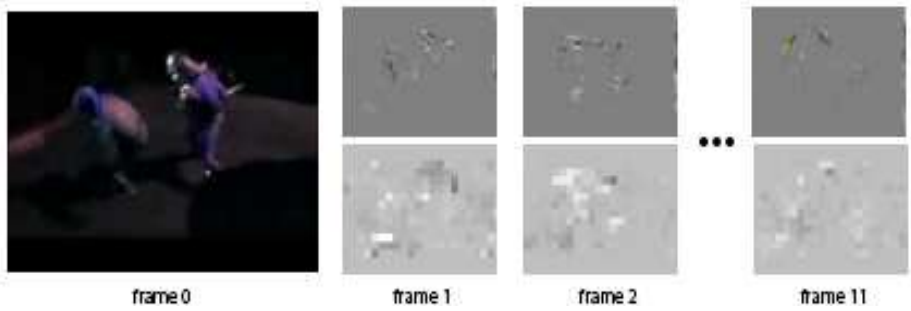

Figure 2. Example of a GOP structure: the *I-frame* is on the left, on the upper row we see the residuals component, on the bottom are the motion vectors frames. The goal of this figure is to build the understanding of the compressed representation and to realize the small amount of data the residual frame consists.

2. Related Work

Video Action Recognition Action recognition in videos is a well-studied task with a variety of datasets [9, 8, 10]. The most common approach to recognition on videos with RGB frames is a two-stream approach [2], which applies convolutional neural networks to separately process RGB frames and optical flow, this method showed significant improvement over previous methods, but holds massive calculations by needing to compute two different networks and by creating optic flow. The individual CNNs have recently used 3D convolution [11] to achieve state-of-the-art results. While 3D convolutions can exploit the temporal domain, prior work still uses optical flow to achieve higher results. Alternative approaches have applied RNNs [12] or other aggregation approaches to reason over the temporal domain.

While providing a significant performance increase, and having a widespread use in action recognition, its high parameter count and computational cost is an important disadvantage. Recently, the efficiency of the operator has been substantially improved by decomposing the 3D convolution filters [13, 14], or employing group convolutions such as in the Multi-Fiber Network [7]. We used Multi-Fiber Network to demonstrate state of the art accuracy without necessarily requiring high computational cost.

Video Compression Video compression is the process of converting digital video into a format suitable for transmission or storage and reduces the number of bits by removing spatial and temporal redundancies. Video codecs splits the video into Group Of Pictures (GOP). The GOP consists of an Intra frame or *I-frame*, which is a self-contained RGB frame with full visual representation, followed by a sequence of inter frames (*P-frames* or *B-frames*) that only represent the changes that occur with respect to the previous frame. Here, we used a GOP with 12 frames (1 *I-frame* and 11 *P-frames* that represent the change in the frames) in order to compare our results with previous works [5, 4, 6].

The *P-frames* hold two types of data: motion vectors and residuals as depicted in Fig. 2), Motion vectors represent the movement of a block of pixels from a source frame to a target frame. In the encoding phase, we predict the motion vectors, and use them in order to warp the source frame. The residual is calculated by subtracting the new frame resulting from the warping and the original frame. The residuals represent the difference between the warped image using the predicted motion vectors and the target frame. In other words, the residual frame holds the error of the motion prediction. For simplicity, we will not use *B-frames*.

Action Recognition and the Compressed Domain Wu *et al.*[4] first proposed to apply deep learning for action recognition directly in the compressed domain with CoViAR. In order to build a frame based model, they used three different 2D CNNs, with each network trained on different types of compressed data: the *I-frame*, and the residuals and motion vectors of the *P-frame*. To boost performance, they also incorporated optical flow, which necessitated a full decoding of the video in order to extract the desired optic flow, which defeats one of the advantages of operating directly in the compressed domain; also it's adding a large capacity load and computation delay as each opticflow frame compute time is between 20-80ms. Other approaches [15, 3] leverage optical flow and motion vectors, which are correlated, to transfer knowledge learned in optical flow.

Specifically, DMC-Net [5] improved on CoViAR by introducing a lightweight generator to reduce motion vector noise and capture fine motion details. This approach does not require calculating optical flow, significantly reducing the computation time. DMC-Net, as Coviar, uses several 2D CNN models each for different modality. Shou *et al.*[5] showed accuracy results replacing their Resnet18 model with the Inflated 3D CNN model from Carreira and Zisserman [11], but that model requires 250 frames at inference time (approximately 8 seconds with FPS of 30), which contradicts our goal of low latency inference and the requirement to perform inference on data in the compressed domain, hence we will not refer to this model. Recently,

Table 1. Comparison with previous works: our model outperforms prior work in the compressed domain while requiring significantly fewer parameters and Flops.

Method	Params.[M]	GFLOPS	UCF-101 Top-1	HMDB51 Top-1
CoViaR [4]	83.6	3,615	90.4	59.1
DMC-net (Resnet18) [5]	95.2	401	90.9	62.8
TTP [**?**]	17.5	1050	87.2	58.2
MFCD-Net (Ours)	**8.0**	**128**	**93.2**	**66.9**

Huo *et al.*[6] followed Coviar [4] and DMC-Net [5] and used also a few 2D CNNs each for every modality, for each 2D CNN they used Mobilenet [16] in order to be more efficient, they introduced a new block named Temporal Trilinear Pooling for combining the three modalities (I-frame, Motion Vectors, Residuals). In summary, all previous works used the same idea with different tweaks; our method is unique and novel in the compressed stream, as explained in the next section.

3. Method

We desire to bring the inference in the compressed domain to a familiar place, the raw video domain; however, instead of having N RGB frames we have only one *I-frame*, the remaining $N-1$ frames are *P-frames*. Our Hypothesis is that the residual component in the *P-frames* can fill the role of the missing RGB frames, and we can process this clip which consist of one *I-frame* and $N-1$ residual frames as one unit using a 3D CNN, the path we are walking in order to achieve it, is described in the next few sections.

3.1. Uncompressed Domain

We first present the notations of the network in the uncompressed domain and this network has two vital roles in our method:

- Measure performance on the uncompressed RGB in order to act as a benchmark.
- A teacher network to the network in the compressed domain (e.g MFCD-Net); it will be denoted as $\Psi_{teacher}$.

We represent each video as a sequence of K frames: $\mathcal{V}_{\mathcal{RGB}} = [X_1, X_2, \ldots, X_K]$, where each frame is an RGB image $X_k \in \mathbb{R}^{H\times W\times C}$. Therefore, the input video will have shape (N, C, K, H, W), where N is the batch size, C number of channels and H, W the spatial size of each frame.

The raw data is fed into $\Psi_{teacher}$, and can be written:

$$y_t = \Psi_{teacher}(\mathcal{V}_{\mathcal{RGB}}) \tag{1}$$

Where y_t is the label predicted, t represents the fact that this is the output of the teacher network.

3.2. Compressed domain

Data modeling Feeding the data into the network is one of the key obstacles in the analysis of videos in the compressed domain.

In the uncompressed domain, every frame is independent, since the content of the k^{th} RGB frame is independent of the $k-1$ RGB frame. However, in the compressed domain, dependencies exist between the frames due to the use of temporal information. Each *P-frame* holds data with respect to the previous frame, which is likely another *P-frame*.

The goal of the data modeling is to make the *P-frames* independent, so we need to make a straight connection between the *I-frame* and each of the *P-frames*. In order to do that, we need to understand which pixels have moved and where.

We will need to trace the motion vectors back to the full image *I-frame*. We are modeling the compressed data similar to CoViAR [4] to remove the dependency on the *I-frame*.

We will summarize the data modeling using our notation. Given a GOP that consists K frames,where the *I-frame* is frame number 0 and the rest (*i.e.*, $[1, K-1]$) are *P-frames*. We'll denote a pixel in a *P-frame* as $n_p = (n_{p,x}, n_{p,y})$, and a pixel in a *I-frame* as :$n_i = (n_{i,x}, n_{i,y})$. Our goal is to connect a pixel in a *P-frame* at time t to a pixel in the *I-frame* n_i, in the *I-frame* at time k, where $k < t$.

The pixel n_p in the residual frame at time $k \in [1, K-1]$ is denoted as $R^k_{n_p}$ and the accumulated residual frame from time step 0 (i.e *I-frame*) to time step k is denoted as $\bar{R}^k_{n_p}$. Although we are not using the motion vectors in our method, it is needed to explain the data modeling. The accumulated motion vectors frame from *I-frame* to time step t is denoted as $\bar{D}^k_{n_p}$. Now, we have a direct connection between each *P-frame* and the *I-frame*, can be written as:

$$U^k_{n_p} = U^0_{n_p - \bar{D}^k_{n_p}} + \bar{R}^k_{n_p} \tag{2}$$

Where $U^k_{n_p}$ is the n_p pixel in the RGB frame at $t = 0$.

Intuitively, equation 2 shows the direct connection between the *P-frame* at time step t to the *I-frame*, by looking a specific pixel in the *I-frame*, moving it to its place in a *P-frame* at time t and adding the accumulated residuals.

In Figure 3, we're able to see the advantages of this method. At time step 3, the residual has almost no data,

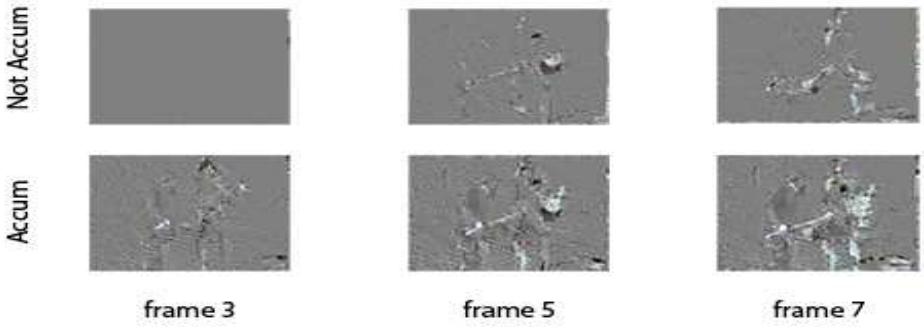

Figure 3. An example of the difference between residuals and accumulated residuals. The accumulate method holds the information through the time steps. Frame 5 shows the effect of previous frames (unlike non-accumulating method), thus we exploit the temporal data.

while the accumulated residual accumulate the new data upon the previous data and give this small amount of data context. Furthermore, we can see time step 5, the residual frame mainly shows one figure in the frame while the accumulated residual shows two which again helps the 3D CNN to understand the full context of each frame.

3.3. Model

In order to propagate features from the *I-frame* to the rest of the residual data in the GOP, we introduce a simple but effective method of accumulated residuals across the time steps. We evaluated several merging techniques and found that the most efficient and accurate method is to simply add the *I-frame* to each residual frame. This helps preserve the change that is expressed by the residual frames but still propagates the scene background, color and texture. Given an *I-frame*, denoted as $U^0 \in R^{HxWx3}$, and accumulated residual frames $\bar{R}^t \in R^{HxWx3}$ for each time step k, we can write the augmented residuals $\hat{R}^k$ as:

$$\hat{R}^k = \frac{(\bar{R}^k + U^0)}{2}, \forall k \in [1, K-1] \tag{3}$$

Proposition 1 *The residual frames (i.e $\hat{R}$) holds enough information to replace the original RGB frames from the original video.*

Once we follow the above proposition, we can use the *I-frame* and the residual frames as one video clip, where instead of 12 RGB frames, the clip holds one *I-frame* and 11 residuals.This enables the use of a 3d convolution network, which showed large improvement in video understanding tasks. Hence our video clip will look as follows:

$$y_s = \Psi_{student}([U^0, \hat{R}^1, ..., \hat{R}^{K-1}]) \tag{4}$$

Where s is to mark that y_s is the prediction for the student network. The augmented residuals operation (i.e adding *I-frame* to each residual frame) is depicted in Fig. 4.

Student Teacher So far we discussed the network in the compressed domain (i.e MFCD-Net) with the data modeling process and the way of looking at the task and data modeling them self bring to state of the art results. However, we desire to take our model a step forward, in order to aid our network in the compressed domain to learn beneficial features that are similar to those in the raw domain (which we desire to be similar to), therefore we use student-teacher framework [17][18]. In this framework the output of a teachers hidden layers and logits are responsible for guiding the student network to predict similar feature maps.

Our notations are as follows: $\Psi^i_{student}$ will represent the the i'th layer in the student network, (i.e the network in the compressed domain), $\Psi^i_{teacher}$ will represent the the i'th layer in the teacher network, (i.e the network in the raw domain). Our new objective function has three ingredients:

1. Hint sharing between the hidden layers, using L_2 distance as presented by Romero *et al.*[17], denoted as:

$$L^i_{hints} = \frac{1}{N}\sum_{n=1}^{N} ||\Psi^i_{student} - \Psi^i_{teacher}||^2 \tag{5}$$

   This objective will force the intermediate feature maps of the student network to look like the intermediate feature maps of the teacher network.

2. KL distance between the soft logits of the two networks as presented by Hinton *et al.*[18]. Let's first denote the soft logits of the student network as: $SL_{student} = Softmax(y^s_n/\tau)$ and of the teacher network: $SL_{teacher} = Softmax(y^t_n/\tau)$ this can be denoted as:

$$L_{SL} = KL(SL_{student}||SL_{teacher}) \tag{6}$$

   Where KL() is the Kullback-Leibler distance.

3. Cross entropy loss between the predicted output and the label, denoted as:

$$L_{CE} = CE(y^s, \tilde{y}) \tag{7}$$

Where $\tilde{y}$ is the ground-truth label and y^s is the predicted label.

The total objective loss is

$$L_{total} = L_{ce} + \lambda_1 \sum_{i=1}^{h} L^i_{hints} + \lambda_2 * L_{SL} \tag{8}$$

Where λ_i indicates the effect of each loss on the total loss function and h is the number of hidden layers. Our full method is depicted in Fig. 1.

4. Ablation Study

In order to understand what is the contribution of each part in our model, we also tested the model's performance when trained on formats different from the previous section.

No Transfer Learning In this experiment, we are not training using the student-teacher (Transfer Learning) framework in order to understand the benefit of this method. A visualization of this experiment can be seen at figure 4.

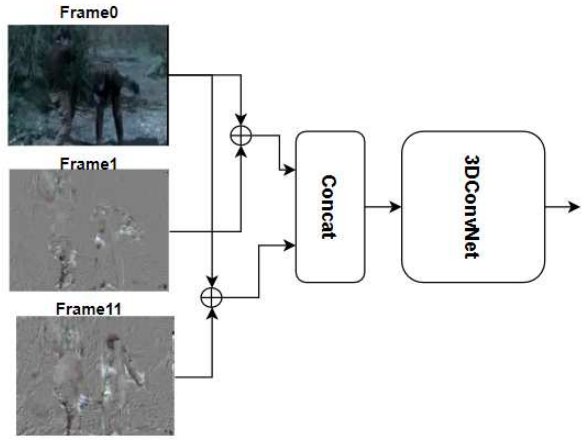

Figure 4. This Figure shows the idea behind adding the *I-frame* to the rest of the residual frames in the GOP

No Augmentation of Residuals. Although the residual frames have little information, we hypothesize that they contain sufficient information for an accurate prediction, without requiring augmentation of the residuals with the *I-frame*, as used in the previous section. We write this as:

$$y = \Psi([U^0, \bar{R}^1, ..., \bar{R}^{11}]) \tag{9}$$

The model can use the solo *I-frame* at the beginning of the GOP as an "anchor" to extract features such as color and texture, that are then implicitly propagated to the residual frames. Figure 5 shows an example of the input GOP, 1 *I-frame* and 11 residual frames.

Residuals Only We also examine the challenging case where our model only has access to the accumulated residuals $\bar{R}$. We can write this method as

$$y = \Psi([\bar{R}^1, \bar{R}^1, \bar{R}^2, ..., \bar{R}^{11}]) \tag{10}$$

In order to keep the number of input frames to 12, we have duplicated the first residual frame once. While this case will never occur in the compressed format of data, its an important ablation study to better understand how 3D CNNs operate in the compressed domain.

5. Experiments

In this section, we discuss the implementation details of our experiments, datasets used, and results achieved.

Datasets We report results on three leading datasets, including HMDB-51[8], UCF-101[9], and Kinetics400 [10]. HMDB-51 contains 6,766 videos from 51 action categories, while UCF-101 contains 13,320 videos from 101 action categories. Both datasets have 3 officially specified training/test splits. Our result are calculated as an average on the three splits. Kinetics dataset holds 400, with 400—1150 clips for each action. The original version has 306,245 videos, and is divided into three sets, training, validation and test. As Kinetics is a list of videos from YouTube, some videos were deleted, hence today out dataset holds around 200k videos for training and 40k videos for validation. Our experiments are done on the validation set.

Training The models were pretrained on Kinetics in the raw domain, then fine-tuned for our task in the compressed domain using cross-entropy loss and SGD optimizer, with a learning rate of 0.005 along with weight decay of 0.0001 and 0.9 for momentum. During training, we split the video into 12-frame clips in order to be aligned with previous works and use the data preparation procedures outlines in the section 3. For the MFNet3D network, each frame in the clip was resized to 256×256 and cropped to a 224×224 frame and horizontally flipped with a 50% probability.

Inference During inference we randomly sampled fifteen 12-frame clips (i.e 1 *I-frame* and 11 residual frame) to generate input clips for our networks, and each clip is randomly cropped. Each clip pass requires 8.53 GFLOPS; therefore, in order to do one prediction we will require 128 GFLOPS.

Table 2. Performance of our full method using residual augmentation and Student Teacher learning compare with the MFNet in the raw domain. This is video level accuracy with 12 frames for each clip, and an average of the 3 splits

	HMDB51			UCF-101		
	RGB	*Ours*	Δ	RGB	*Ours*	Δ
MFCD-Net (Ours)	69.6	66.9	2.7	94.1	93.2	0.9

Our advantage over previous work is the fact that we are seeing the residuals as a substitute of the missing RGBs and not a different modality, which allows us to use only one network and not multiple 2D CNNs.

Our experiment section has four components:

- Compare the accuracy of our full method (i.e MFCD-Net) with the RGB raw domain accuracy using MFNet.
- Compare our MFCD-Net with other approaches in the compressed domain.
- Extensive ablation study.
- Kinetics experiments comparison with raw domain networks.

5.1. Accuracy comparison: compressed *vs.* uncompressed domain

We applied our method on MultiFiberNet3D [7]. The model was pre-trained on the Kinetics [10] raw data. Then it was fine tuned on leading datasets: HMDB51 [8] and UCF101 [9] raw data in order to be our reference benchmark. The same pre-trained models were also fine tuned on HMDB51, UCF101 and Kinetics compressed data representation (as described in Section 3.2).

As shown in Table 2, the difference in performance between using the raw RGB videos and the video representation in the compressed domain are small, and our method takes a significant step towards making the work in the compressed domain more practical, showing less than 1 top1 decrease in UCF101 dataset and a decrease of 2.7 points in accuracy in HMDB51.

5.2. Comparison with State-of-the-Art

In the field of Action Recognition in the compressed domain, there are three leading models: CoViAR [4], DMC-Net [5] and TTP [6]. We compare our model in terms of accuracy (Top-1), computational requirements (GFLOPS) and memory requirements (number of parameters). Coviar [4] test procedure is as follows: from each video they sampled 25 *I-frames*, 25 residual frames, and 25 motion vector frames, with each frame is getting 2 different flips and 5 different crops for data augmentation [6], and followed the settings of Wu *et al.*[4]. DMC-Net [5] partially followed Coviar settings [4], except the data augmentation part. Those important details must be considered when attempting to solve this task, as efficiency is a major criterion.

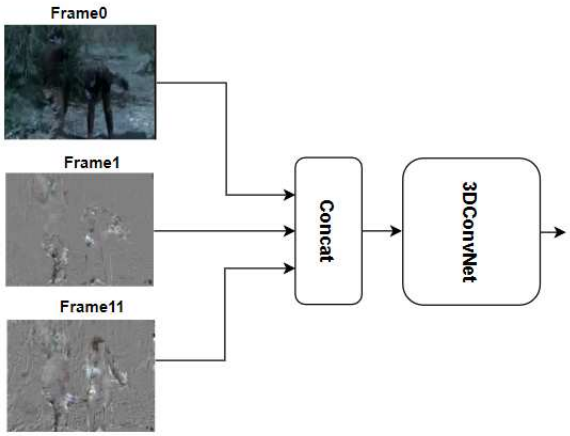

Figure 5. This figure shows the basic configuration of feeding a group of picture in the compressed domain into a 3DCNN

Computation Evaluation Regarding computational requirements in video action recognition, we need to examine how many GFLOPS are needed for one prediction (which consist of multiple data passes). Previous works presented the average GFLOPS required for processing each frame; even if we presented the GFLOPS as previous works did, per frame, our method consume 8.53 GFLOPS for each clip hence each frame is $8.53/12 = 0.71$ GFLOPS (which is the most efficient method). But calculating GFLOPS per frame losses any correlation to the accuracy reported, since it depends on the test procedure, hence we will report the GFLOPS required to infer 1 video. Our method samples 15 clips from each video, then our total amount of GFLOPS is: $8.53 * 15 = 128$. For example, TTP [6] claims to require 1.4 GFLOPS, which is the average number of required operations for passing 1 frame through MobileNetV2 [16], but their inference procedure is the same as Coviar, which means that they are sampling 75 frames and for each frame they are doing 10 augmentations, so there are 750 passes through Mobilenet, which sums up to $750 * 1.4 = 1050$ for a single video inference.

As shown in Table 1, our model significantly outperforms the previous state-of-the-art. In HMDB51, our accuracy is higher by 4.1 points from the current state-of-the-art, 7.8 from Coviar and 8.7 top1 points from TTP. In UCF101, our accuracy is higher by 2.3 top1 points from the state-of-the-art DMC-Net, 2.7 from CoViar and 6 top1 points higher than TTP. With the impressive leap in accuracy mentioned

Table 3. Performance for our ablation study as mentioned in Section 5.3: for HMDB51 and UCF101 datasets, performance was measured on split1.

Model	Format	HMDB51	UCF101
MFCD-Net	Full Method	66.9	93.2
MFCD-Net	I + Agmt. Residuals $\hat{R}$	64.8	92.4
MFCD-Net	I + Residuals $\bar{R}$	64.4	92.5
MFCD-Net	Residuals only $\bar{R}$	60.1	89.0

Table 4. Top-1 accuracy on Kinetics val. set: results for other models are from Table 2 of Hara *et al.*[19] and are using the uncompressed RGB data; our model uses the compressed representation.

Model	Kinetics Top-1
DenseNet-201	61.3
ResNet-152	63.0
ResNet-200	63.1
Wide ResNet-50	64.1
ResNext-101	65.1
MFCD-Net (Ours)	**68.3**

above, it's expected that requirements should be massive but our model is more efficient than previous work as we no longer need to keep three or four 2D CNNs, we no longer need heavy augmentation process and our departure from the motion vectors helps us with efficiency. Our experiments shows that our method requires 68% less GFLOPS and 92% reduction in memory footprint than the state-of-the-art DMC-Net. Model size is critical for performance, as it allows the model to be cached on an accelerator's local memory, leading to a significant increase in performance.

5.3. Ablation Study Results

In this section, we discuss the importance of each and every element of our method, using the results from the experiments presented in Table 3. We see that the student-teacher framework increased the performance drastically, which helps us to get closer to the raw domain. We also say that the Augmented Residuals phase, where we add *I-frame* to the residual frames in the GOP holds limited contribution, if any, to top1 accuracy. When looking at the last row we can understand how important is the first and only *I-frame* in the compressed clip; more than 4 top1 points decreased when we dropped the *I-frame* from the clip.

5.4. Kinetics

We trained our model on the Kinetics400 training set (raw domain). We then fine-tuned our network on Kinetics (compressed domain) and evaluated on Kinetics validation set (compressed domain). We compared our results to models presented in [19], evaluated on the Kinetics validation set. Those models were trained and evaluated in the raw domain. Our model significantly beats those models.

6. Discussion

One might claim that our use with 3dCNN made our results much better, that is absolutely correct, Our contribution was in suggesting the hypothesis, which residual frames can replace RGB frame and suggesting a piratical way of doing it, this novelty enabled us to use 3dCNNs which are much more suitable for videos instead of 2dCNNs as used in previous works. Video is a redundant data type. It effects our ability to analyze this data type as it requires us massive calculations and storage, with little ROI. One of the ways to reduce this redundancy and reduce storage massively is to work in the compressed domain. As shown above, video compression is the best redundancy exploiter that can be found. The primary savings by using residual frames comes from their small memory footprint. Further work should focus on reducing the model size, since now the video consists less data than the RGB video clip.

7. Conclusions

In this paper, we introduce a novel way to examine the task of action recognition in the compressed domain. Different from previous research, we claim that the components of the compressed stream can be seen as a unified entity and that the residual frames are able to supplant the omitted RGB frames and form a pseudo raw video clip along with the *I-frame* at the beginning of the GOP. Our method did not use motion vectors from the compressed stream (in order to save storage and computation), nevertheless we achieved state-of-the-art results beating previous research [4, 5, 6] by a large gap, with dramatically fewer parameters and lower computation requirements. Furthermore, we achieved minimal degradation in performance compared to the same architecture RGB frames [19, 7], which is the main purpose of this research (*i.e.*, to achieve the accuracy of a network in the raw domain working in the compressed domain).

Although the field of action recognition in the compressed domain has a few practical issues, it holds great potential to more efficient inference and training. Our method and experiments are a step forward towards enabling efficient deployment of video recognition in all platforms.

References

[1] "Cisco Visual Networking Index: Forecast and Trends, 20172022 White Paper - Cisco," 2019. 1

[2] K. Simonyan and A. Zisserman, "Two-Stream Convolutional Networks for Action Recognition in Videos," in *Neurips*, pp. 568–576, 2014. 1, 3

[3] B. Zhang, L. Wang, Z. Wang, Y. Qiao, and H. Wang, "Real-Time Action Recognition With Deeply Transferred Motion Vector CNNs," *IEEE Trans. Image Process.*, vol. 27, pp. 2326–2339, may 2018. 1, 3

[4] C.-Y. Wu, M. Zaheer, H. Hu, R. Manmatha, A. J. Smola, and P. Krahenbuhl, "Compressed Video Action Recognition," in *2018 IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, pp. 6026–6035, IEEE, jun 2018. 1, 2, 3, 4, 7, 8

[5] Z. Shou, Z. Yan, Y. Kalantidis, L. Sevilla-Lara, M. Rohrbach, X. Lin, and S.-F. Chang, "DMC-Net: Generating Discriminative Motion Cues for Fast Compressed Video Action Recognition," tech. rep., Columbia Univ. & Facebook, 2019. 1, 2, 3, 4, 7, 8

[6] Y. Huo, X. Xu, Y. Lu, Y. Niu, Z. Lu, and J.-R. Wen, "Mobile Video Action Recognition," tech. rep., 2019. 1, 2, 3, 4, 7, 8

[7] Y. Chen, Y. Kalantidis, J. Li, S. Yan, and J. Feng, "Multi-Fiber Networks for Video Recognition," in *ECCV*, 2018. 2, 3, 7, 8

[8] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "HMDB: A large video database for human motion recognition," in *2011 Int. Conf. Comput. Vis.*, pp. 2556–2563, IEEE, nov 2011. 2, 3, 6, 7

[9] Khurram Soomro, A. R. Zamir, and M. Shah, "UCF101: A Dataset of 101 Human Actions Classes From Videos in The Wild," tech. rep., University of Central Florida, 2012. 2, 3, 6, 7

[10] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, M. Suleyman, and A. Zisserman, "The Kinetics Human Action Video Dataset," tech. rep., Google, 2017. 3, 6, 7

[11] J. Carreira and A. Zisserman, "Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset," tech. rep., Google, 2017. 3

[12] J. Donahue, L. A. Hendricks, M. Rohrbach, S. Venugopalan, S. Guadarrama, K. Saenko, and T. Darrell, "Long-Term Recurrent Convolutional Networks for Visual Recognition and Description," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, pp. 677–691, apr 2017. 3

[13] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri, "A Closer Look at Spatiotemporal Convolutions for Action Recognition," in *2018 IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, pp. 6450–6459, IEEE, jun 2018. 3

[14] S. Xie, C. Sun, J. Huang, Z. Tu, and K. Murphy, "Rethinking Spatiotemporal Feature Learning: Speed-Accuracy Trade-offs in Video Classification," in *ECCV*, pp. 318–335, Springer, 2018. 3

[15] B. Zhang, L. Wang, Z. Wang, Y. Qiao, and H. Wang, "Real-time Action Recognition with Enhanced Motion Vector CNNs," in *CVPR*, 2016. 3

[16] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 4510–4520, IEEE Computer Society, dec 2018. 4, 7

[17] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio, "FitNets: Hints for Thin Deep Nets," in *ICLR*, dec 2015. 5

[18] G. Hinton, O. Vinyals, and J. Dean, "Distilling the Knowledge in a Neural Network," in *NIPS 2014 Deep Learn. Work.*, mar 2014. 5

[19] K. Hara, H. Kataoka, and Y. Satoh, "Can Spatiotemporal 3D CNNs Retrace the History of 2D CNNs and ImageNet?," in *2018 IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, pp. 6546–6555, IEEE, jun 2018. 8